



MICRO-50, Boston, October 2017

Tutorial:

Microarchitecture Level Reliability Assessment

Throughput and Accuracy

<http://micro50-tutorial.di.uoa.gr/>

Organizers/Presenters:

Athanasios Chatzidimitriou, Manolis Kaliorakis, Dimitris Gizopoulos

Tutorial Outline

- **Introduction:** Microarchitecture-Level Reliability Assessment
- **GeFIN** Fault Injection Engine and Fast Modes
- **MeRLiN** Fault Classification
- **Use Cases**, Results
- **Discussion**
- **Closing**



MICRO-50, Boston





MICRO-50, Boston, October 2017

Part 1:

Introduction to Microarchitecture Level Reliability Assessment

Dependability

- **Dependability***:
 - (a) the ability to deliver service that can justifiably be trusted, or
 - (b) the ability to avoid service failures that are more frequent and more severe than is acceptable.

* “Basic Concepts and Taxonomy of Dependable and Secure Computing”, A.Avizienis, J.-C.Laprie, B.Randell, C.Landwehr, IEEE Transactions on Dependable and Secure Computing, vol. 1, no. 1, January-March 2004



MICRO-50, Boston



Reliability

- Dependability encompasses the following attributes:
 - **availability**: readiness for correct service
 - **reliability***: continuity of correct service
 - **safety**: absence of catastrophic consequences on user(s) and environment
 - **integrity**: absence of improper system alterations
 - **maintainability**: ability to undergo modifications/repairs

* “Basic Concepts and Taxonomy of Dependable and Secure Computing”, A.Avizienis, J.-C.Laprie, B.Randell, C.Landwehr, IEEE Transactions on Dependable and Secure Computing, vol. 1, no. 1, January-March 2004



MICRO-50, Boston



Fault Models

- **Soft error** (transient fault)
 - A bit is **temporarily** flipped (0->1 or 1->0)
- **Hard errors** (permanent faults)
 - A bit is **permanently** stuck at 0 or 1
- **Intermittent errors**
 - A bit is stuck at 0 or 1 for a **certain interval**
- **Multi-bit faults**
 - Temporal or Spatial



Terminology

- **SER – Soft Error Rate**
 - rate at which soft errors are encountered (measured) or predicted to be encountered
- **FIT – Failures-in-Time**
 - number of failures per 1 billion hours of operation
- **MTTF – Mean Time To Failure**
 - $1/\text{FIT}$
- **1 FIT ~ 114K years MTTF or**
- **1 year MTTF ~ 114K FIT**
- **AVF – Architectural Vulnerability Factor [0...1]**
 - of a microprocessor structure – probability that a soft error in a bit of the structure affects program execution – depends on the microarchitecture (hardware) and the program (software)



CPU FIT Calculation

- **FIT (Failures in Time)**
 - number of failures of a system in 10^9 hours of operation

- **For each Structure**

$$\text{FIT}_{\text{STRUCTURE}} = \text{FIT}_{\text{BIT}} \times \text{AVF}_{\text{STRUCTURE}} \times \text{\#Bits}_{\text{STRUCTURE}}$$

technology microarchitecture size
+ software

- **For the entire CPU (additive)**

$$\text{FIT}_{\text{CPU}} = \text{FIT}_{S_1} + \text{FIT}_{S_2} + \dots + \text{FIT}_{S_n}$$

Example CPU FIT Calculation

How to measure?

Structure	Raw FIT/bit	Size (bits)	$AVF_{STRUCTURE}$	$FIT_{STRUCTURE}$
A	0.001	1K	0.08	$FIT_A = 0.082$
B	0.001	64K	0.12	$FIT_B = 7.864$
C	0.01	32K	0.02	$FIT_C = 6.554$
D	0.01	8K	0.17	$FIT_D = 13.926$
E	0.01	4K	0.24	$FIT_E = 9.830$
F	0.001	256K	0.11	$FIT_F = 28.836$
				$FIT_{CPU} = 67.092$



Reliability Costs

- Costs: area, performance, power, effort
- Protection: **detect**, **diagnose**, **recover**, **repair**
- Example: memory protection area:

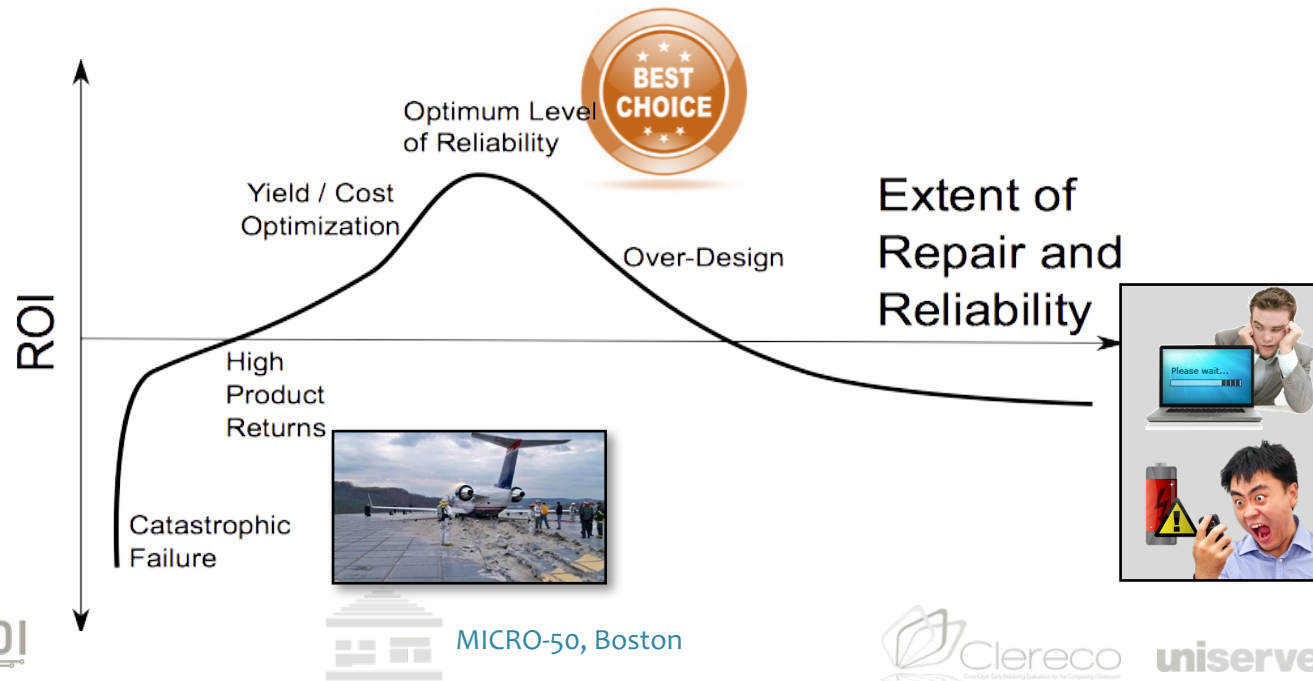
Memory Protection	Extra Storage
Parity	1.56%
SEC-DED	12.50%
DEC-TED	23.40%
Chipkill (IBM) DDDC (Intel)	12.50%
RAIM (zEnterprise)	40.60%
Mirroring (POWER7 RAS)	125.00%



* Y.Luo, *et. al.* "Characterizing Application Memory Error Vulnerability to Optimize Datacenter Cost via Heterogeneous-Reliability Memory", DSN 2014.

Balanced Reliability Provision

- **High costs associated with**
 - **“Too much reliability”**,
i.e. over-designed products – delay, power, area
 - **“Too little reliability”**,
i.e. unreliable products – field returns, reputation



Reliability Assessment – Options

- Level
 - Microarchitecture Level – **early available**
 - Register Transfer Level (RTL) – **too late**
- Simulation speed
 - Microarchitecture: **100x-1000x faster** than RTL*
- Method
 - Massive injections – **accurate but slow**
 - One-time analytical – **fast but pessimistic**

* S.Raach, A.Biswas, J.Stephan, P.Racunas, J.Emer, “A Fast and Accurate Analytical Technique to Computer the AVF of Sequential Bits in a Processor”, ACM/IEEE MICRO 2015.

Speed of Injection @Different Layers

Abstraction Layer	Performance (cycles/sec)
Software	3×10^9
Architecture (simulation)	6×10^7
Microarchitecture (simulation)	3×10^6 (simple CPU) 2×10^5 (detailed CPU)
Flip-flop (simulation)	6×10^2

↓ 3+ orders slower

* H.Cho, S.Mirkhani, C.-Y.Cher, J.A.Abraham, S.Mitra, "Quantitative Evaluation of Soft Error Injection Techniques for Robust System Design", ACM/IEEE DAC 2013.

The Objective



- Reliability Assessment



Early

Microarchitecture
Level

(available early)



Accurate

Massive
injections

*(statistical
significance)*



Fast

Maximize
Throughput/Speed

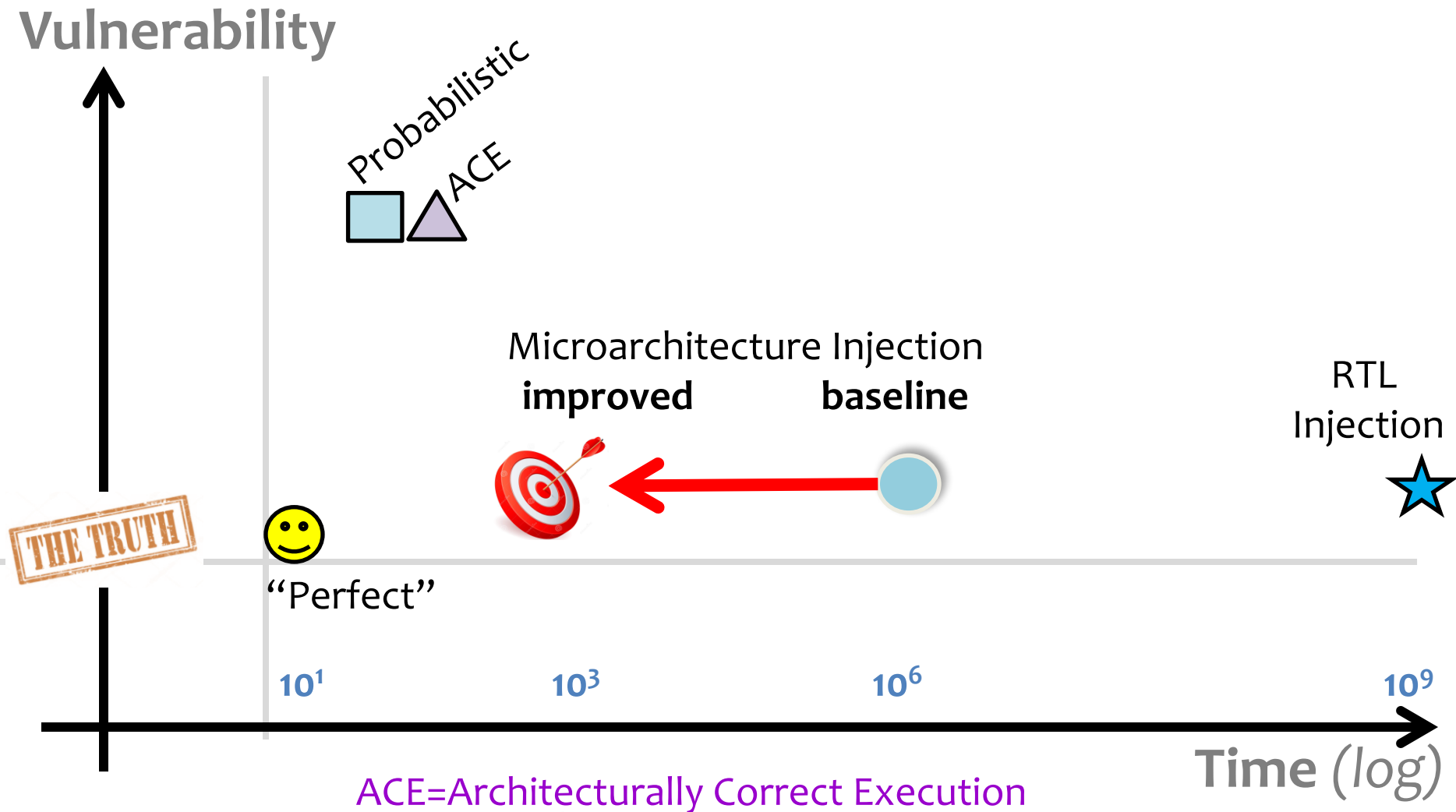
**(our GeFIN and
MeRLiN tools)**

ACE Analysis and Injection

ACE Analysis	Fault Injection
AVF= 1.00	AVF=?
Record ACE and unACE (a) intervals ("useful" bits read by instructions)	Select set of faults
Identify Dead Instructions (b) ("useful" bits are not used)	Inject fault + Run program
Measure Logical Masking (c) ("useful" bits are masked)	Log outcome (output, exceptions, detections, crashes, ...) – REPEAT
$AVF_{ACE} = 1.00 - a - b - c$	$AVF_{FI} = \text{failures/runs}$

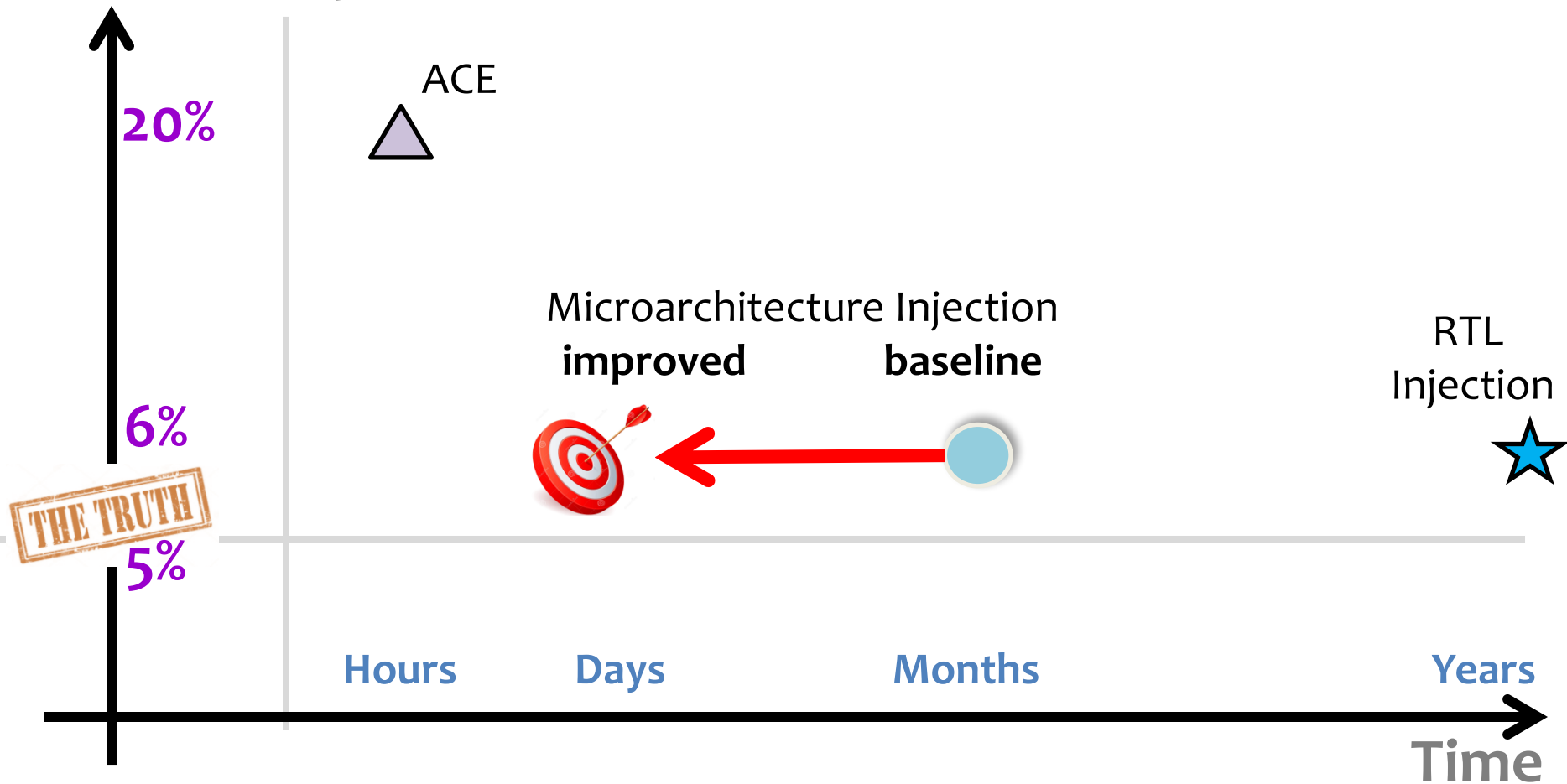


Speed vs. Accuracy

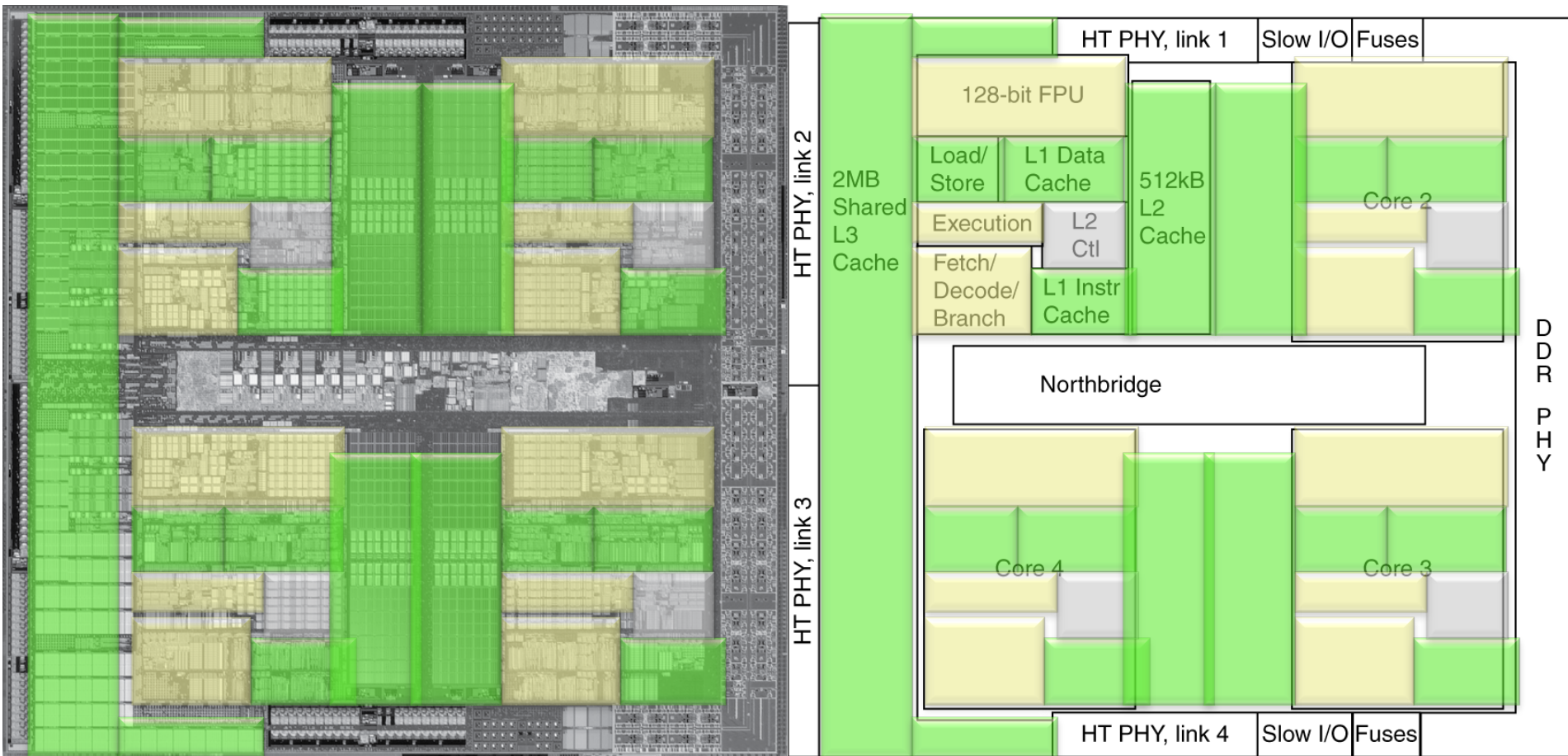


Speed vs. Accuracy – example (1 program + 1 hardware structure)






Vulnerability



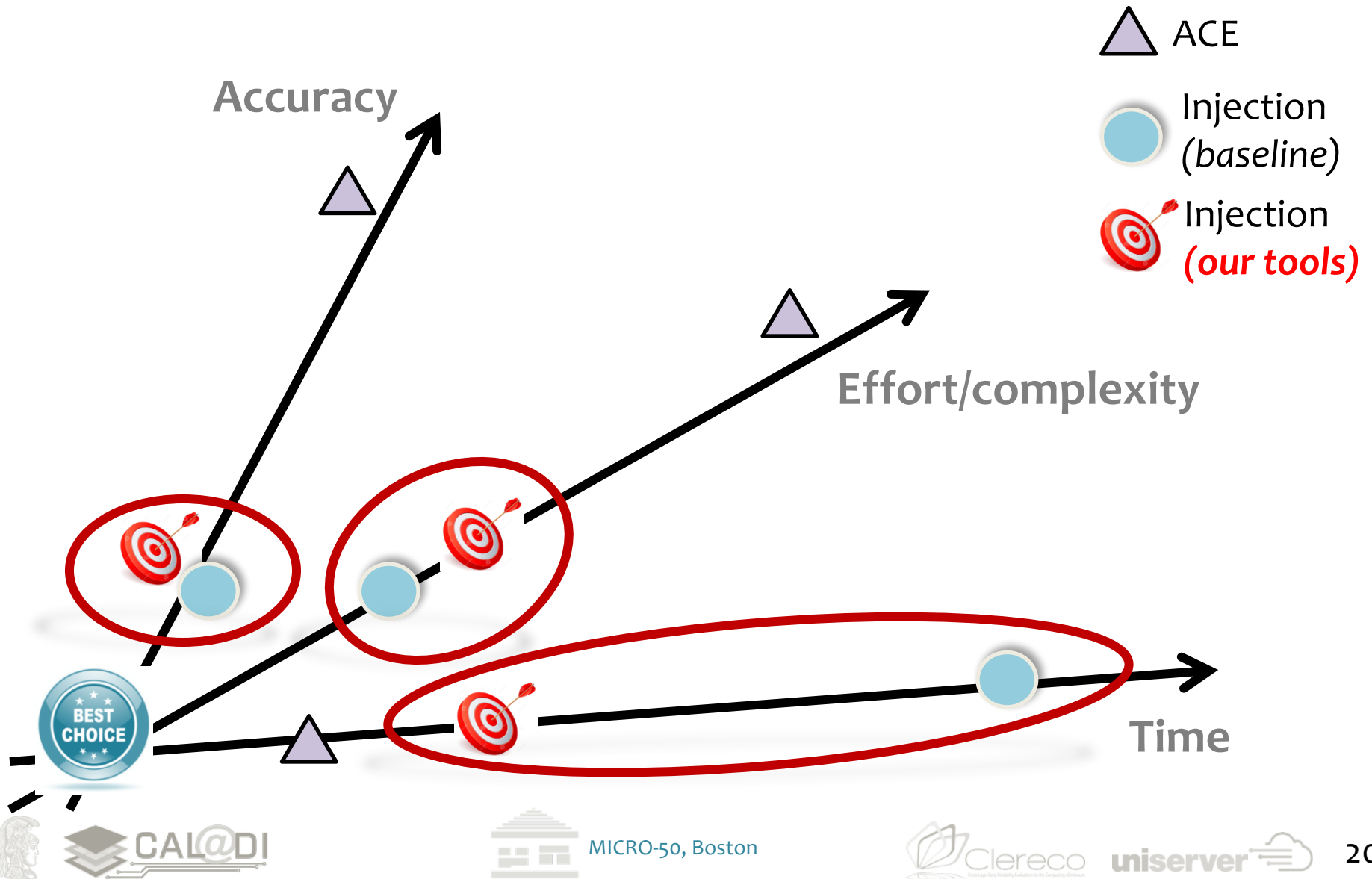
Microprocessor Estate Modeled



Hardware Components at Microarchitecture Level

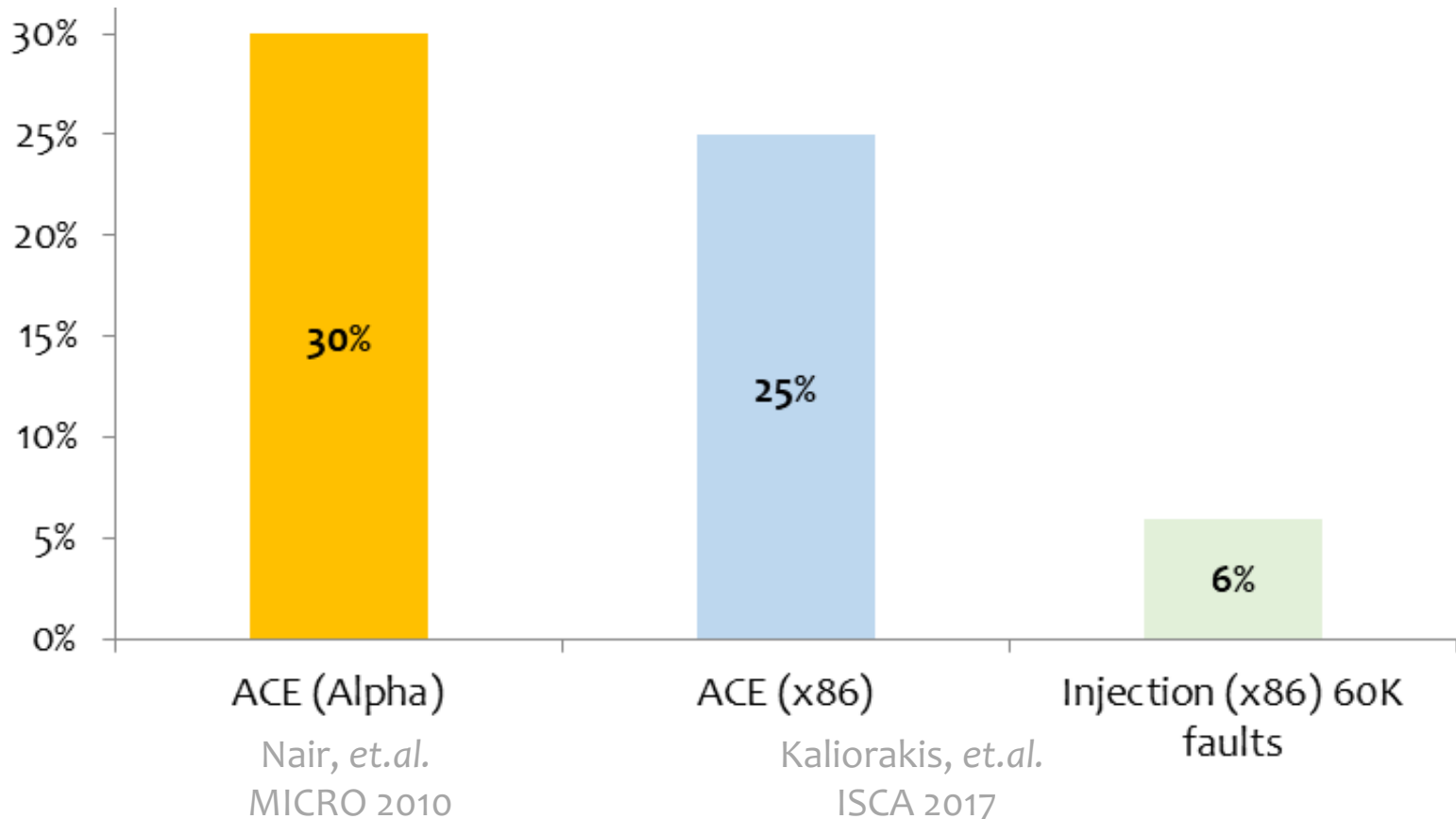
Hardware Component Class	Reliability Measured at Microarchitecture Simulators
Register Files	
Instruction Queues	
Load/Store Queues	
Caches	
DRAM	
Sequential bits	
Combinational logic	

Microarchitecture-Level Reliability: Injections vs. ACE



Injection vs. ACE

- Reg.file AVF (80 regs); same MiBench progs.



Statistics of Fault Injection

- How many Injections?
- **Statistical Fault Sampling (SFI)***
 - Number of injections (n) depends on:
 - initial population (N) = #bits x #clock-cycles
 - statistical confidence level (t)
 - statistical error margin (e)

$$n = \frac{N}{1 + e^2 \times \frac{N - 1}{t^2 \times p \times (1 - p)}}$$

* Leveugle, et. al., "Statistical Fault Injection: Quantified Error and Confidence", DATE, 2009

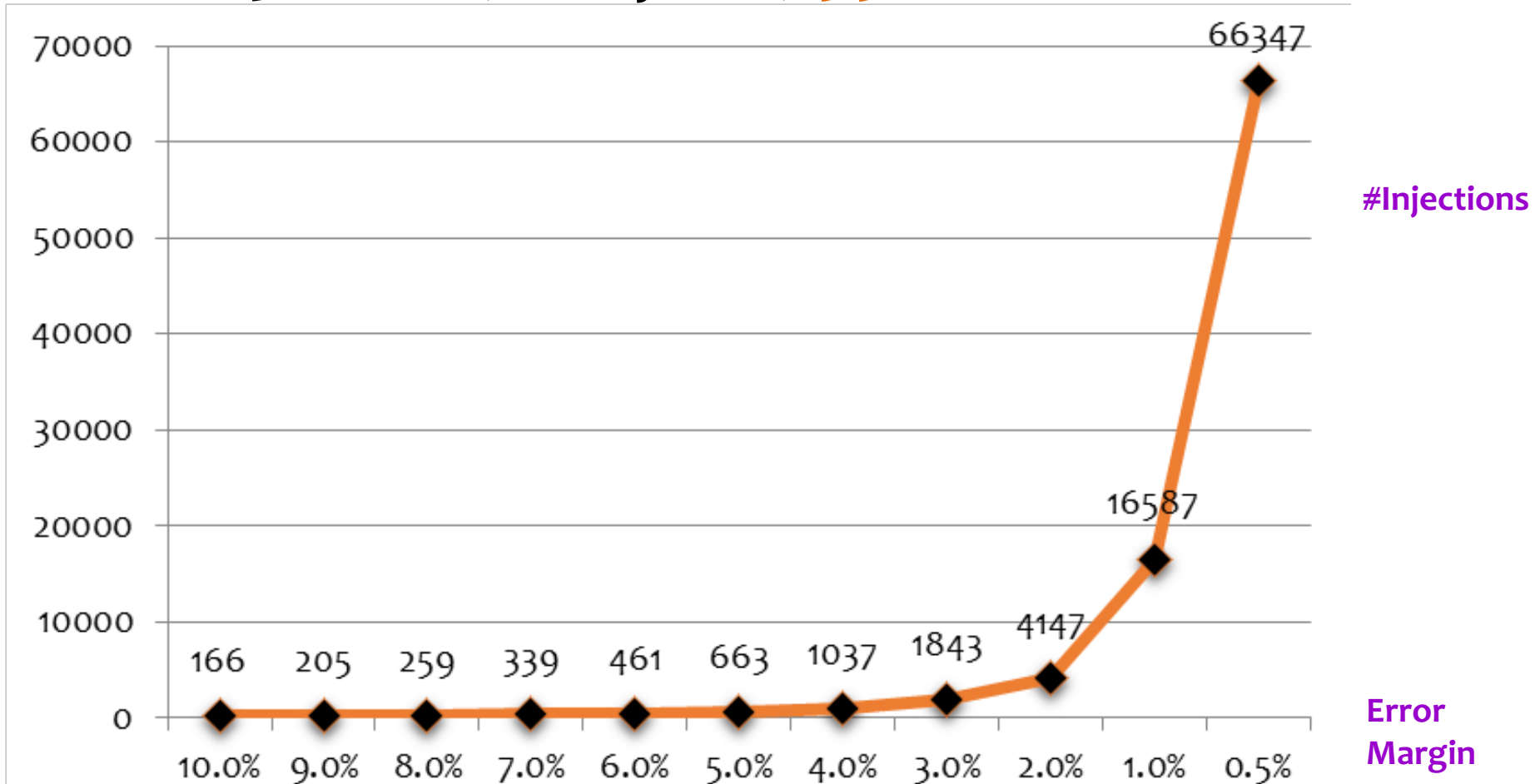


MICRO-50, Boston



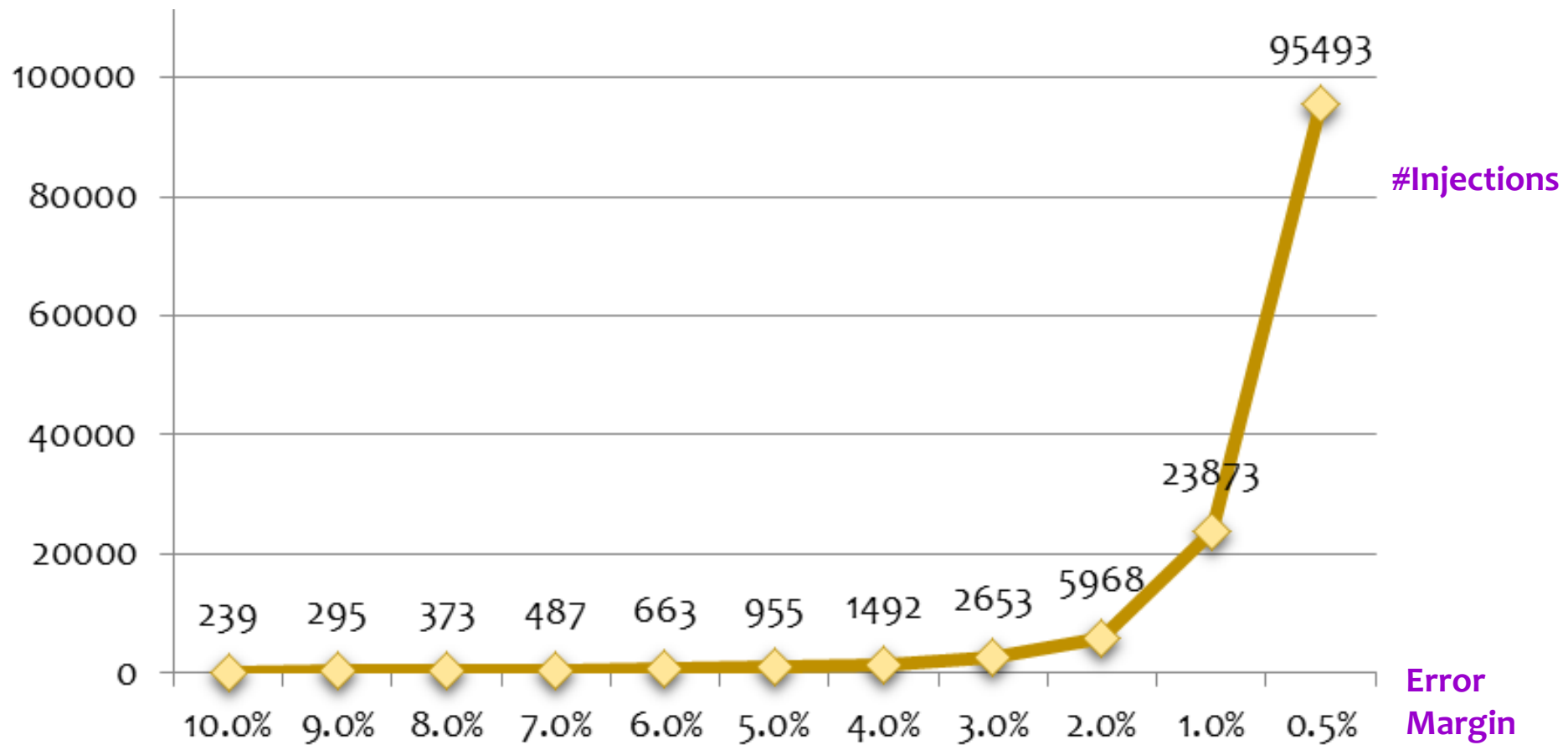
#Fault Injections vs. Error Margin

- For 32K bits, 1B cycles, **99%** confidence



#Fault Injections vs. Error Margin

- For 32K bits, 1B cycles, **99.8%** confidence



Injection Throughput Calculation

- Hardware + Software

parameter	value
program execution time	1B (10^9) dynamic instructions
hardware structure	10K bits (reg.file)
simulation throughput	300K instructions/sec
equipment	10 servers

- Injection Statistics

confidence, error margin		#Injections	Fault injection campaign time
95%	5%	384	1.5 day
99%	3%	1843	1 week
99.8%	1%	23,873	3 months
99.8%	0.5%	95,493	1 year

Needs improvement 😊

Our Tools to Speed Up Injections

GeFIN
Baseline
Fault
Injection
Engine

GeFIN = Gem5-based Fault Injector

MeRLiN = Microarchitectural evaluation
of Reliability using statistical fault iNjection

GeFIN
Fast Modes

MeRLiN
Fault
Pruning



MICRO-50, Boston, October 2017

Next ...
Part 2

Tutorial:

Microarchitecture Level Reliability Assessment

Throughput and Accuracy
<http://micro50-tutorial.di.uoa.gr/>

Organizers/Presenters:

Athanasios Chatzidimitriou, Manolis Kaliorakis, Dimitris Gizopoulos